AMENDMENTS TO THE CLAIMS

1. (Currently amended) A computer-implemented method for use in emulating a processor of a first type which observes a first convention for ordering the significance of bytes within words on a second type of processor which observe a second convention for ordering the significance of bytes within words, comprising:

receiving instructions for execution by said processor of the first type, wherein the instructions comprise memory access addresses in which a byte order in accordance with the first convention is observed;

transforming the memory access addresses for an entire address space ~~of~~ over a plurality of words such that bytes stored in a memory addressed by ~~a~~ the processor of the second type as a result of ~~an~~ the instructions ~~in which a byte order in accordance with the first convention is observed~~ are distributed in a pattern which is a mirror image of the distribution pattern of the bytes which would result if the memory was addressed by ~~a~~ the processor of the first type in response to the said instructions; and,

issuing instructions for execution by said processor of the second type including said transformed memory access addresses.

2. (Currently amended) A computer-implemented method for use in emulating a processor of a first type which observes a first convention for ordering the significance of bytes within words on a second type of processor which observes a second convention for ordering the significance of bytes within words, the order of the second convention being the reverse of the order of the first, comprising:

receiving instructions for execution by said processor of the first type, wherein the instructions comprise memory access addresses in which a byte order in accordance with the first convention is observed;

transforming the memory access addresses such that:

(a) the offset between addresses of any two bytes stored in memory is unaltered by the transformation, wherein said any two bytes are spaced apart in memory by a plurality of words; and

(b) the relative order of the addresses of said any two bytes stored in the memory is reversed by the transformation; and,

issuing instructions for execution by said processor of the second type including said transformed memory access addresses.

3. (Canceled)

4. (Currently amended) A computer-implemented method for use in emulating a processor of a first type which observes a first convention for ordering the significance of bytes within words on a second type of processor which observes a second convention for ordering the significance of bytes within words, comprising:

receiving instructions for execution by said processor of the first type, wherein the instructions comprise memory access addresses in which a byte order in accordance with the first convention is observed;

transforming a plurality of the memory access addresses relating to a plurality of words such that each memory access address B of string length L is transformed to the address A-B-L+S, wherein A is the total number of bytes allocated to a program, and S is the start address of the program; and,

issuing instructions for execution by said processor of the second type including said transformed memory access addresses.

5. (Previously presented) A process for compiling or translating a computer program code instruction using transformed address space references in the compiled or translated code especially configured for execution on a programmable machine utilizing a corresponding predetermined convention for ordering the significance of bytes within words of said address space, said process comprising:

(a) during compilation or translation of a code instruction referring to a memory address, transforming the referenced memory address with respect to a fixed block size of a plurality of words in memory in the predetermined programmable machine so as to change the referenced address value by an amount that is fixed for a given number of bytes being accessed in each word; and

(b) including the thus changed address reference in a compiled or translated output instruction so that there is no extra operation required during execution of the output instruction to accommodate the convention for ordering bytes within words used by said predetermined programmable machine.

6. (Original) A process according to claim 5, wherein said code is a computer program source code.

7. (Original) A process as in claim 5, wherein said change causes said fixed block of memory to be addressed from a predetermined one of its two ends depending upon the convention utilized by said predetermined programmable machine for ordering the significance of bytes within words.

8. (Original) A process as in claim 5, wherein said change causes the fixed block of memory contents for a big-endian machine to be inverted to the mirror image of that for a little-endian machine.

9. (Previously presented) An endian transformation system, comprising:

a memory having a plurality of memory access addresses to address a plurality of words; and

a second type of processor for accessing said memory using said memory access addresses, having means for transforming memory access addresses for use when emulating a processor of a first type which observes a first convention for ordering the significance of bytes within words on said second type of processor which observes a second convention for ordering the significance of bytes within words,

wherein memory access addresses for an entire address space are transformed for a plurality of words such that bytes stored in said memory addressed by said second type of processor as a result of an instruction in which a byte order in accordance with the first convention is observed are distributed in a pattern which is a mirror image of the distribution pattern of the bytes which would result if the memory was addressed by a processor of the first type in response to the said instruction.

10. (Previously presented) An endian transformation system, comprising:

a memory having a plurality of memory access addresses to address a plurality of words; and

a second type of processor for accessing said memory using said memory access addresses, having means for transforming memory access addresses for use when emulating a processor of a first type which observes a first convention for ordering the significance of bytes within words on said second type of processor which observes a second convention for ordering the significance of bytes within words, the order of the second convention being the reverse of the order of the first,

wherein memory access addresses are transformed such that

(a) the offset between addresses of any two bytes stored in memory is unaltered by the transformation, wherein said any two bytes are spaced apart in memory by a plurality of words and

(b) the relative order of the addresses of said any two bytes stored in the memory is reversed by the transformation.

11. (Canceled)

12. (Previously presented) An endian transformation system, comprising:

a memory having a plurality of memory access addresses to address a plurality of words; and

a second type of processor for accessing said memory using said memory access addresses, having means for transforming memory access addresses relating to said plurality of words for use when emulating a processor of a first type which observes a first convention for ordering the significance of bytes within words on said second type of processor which observes a second convention for ordering the significance of bytes within words, wherein each memory access address B of string length L is transformed to the address A-B-L+S, wherein A is the total number of bytes allocated to a program, and S is the start address of the program.

13. (Canceled).

14. (New) A computer-implemented process to enable code instructions adapted for execution on a first type of processor which observes a first endian format for ordering the significance of bytes within words to be executed by a second type of processor which observes a second endian format for ordering the significance of bytes within words, comprising the steps of:

(a) in a translation phase:

allocating a memory address range of length A bytes comprising a plurality of words arranged in a first relative order with respect to a starting address S;

receiving code instructions having memory access addresses which address the memory address range according to the first endian format for ordering the significance of bytes within words, where each access address B is of string length L;

transforming said memory access addresses into a respective transformed memory access address A-B-L+S, such that the relative order of bytes within each word is reversed into the second endian format and the plurality of words are addressed in a second relative order with respect to the given starting address which is a mirror image reverse of the first relative order; and

translating the code instructions into output code instructions executable by the second type of processor, where said output code instructions include said transformed memory access addresses; and,

(b) in an execution phase:

executing said output code instructions on said second type of processor to fetch and store data in the plurality of words in the memory address range, using the transformed memory access addresses.

15. (New) The method of claim 14, comprising

in the translation phase, folding constant terms in the expression A-B-L+S to provide partially resolved transformed memory access addresses in the translated output code instructions; and,

in the execution phase, calculating variable terms of the expression A-B-L+S to provide fully resolved transformed memory access addresses.